KRYLOV SUBSPACE ITERATIONS FOR DETERMINISTIC k-EIGENVALUE CALCULATIONS

James S. Warsa, Todd A. Wareing, Jim E. Morel, John M. McGhee
Transport Methods Group
Los Alamos National Laboratory
Los Alamos, NM 87545-0001

and

Richard B. Lehoucq
Computational Mathematics and Algorithms Department
Sandia National Laboratories
Albuquerque, NM 87185-1110

Send proofs to:

James S. Warsa
Los Alamos National Laboratory
CCS-4, MS D409
Los Alamos, NM 87545-0001

E-mail: *warsa@lanl.gov*
FAX: 505-665-5538

Number of pages:	21
Number of figures:	4
Number of tables:	4

KRYLOV SUBSPACE ITERATIONS FOR DETERMINISTIC k-EIGENVALUE CALCULATIONS

James S. Warsa, Todd A. Wareing, Jim E. Morel, John M. McGhee
Transport Methods Group
Los Alamos National Laboratory
Los Alamos, NM 87545-0001

and

Richard B. Lehoucq
Computational Mathematics and Algorithms Department
Sandia National Laboratories
Albuquerque, NM 87185-1110

Abstract

We discuss the application of the Implicitly Restarted Arnoldi Method (IRAM), a Krylov subspace iterative method, for calculating the k-eigenvalues and corresponding eigenvectors of of criticality problems using deterministic transport codes. An computationlly efficient alternative to the power iteration method that is typically used for such problems, the IRAM can not only calculate the largest eigenvalue, but also several additional higher order eigenvectors, with little extra computational cost. Furthermore, implementation requires only modest changes to the existing power iteration schemes already present an S_N transport code. We present numerical results for three dimensional S_N transport on unstructured tetrahedral meshes to compare the IRAM results with those computed using the traditional, unaccelerated power iteration method. The results indicate that the IRAM can be an efficient and powerful technique, especially for problems with dominance ratios approaching unity.

1 INTRODUCTION

The dominant k-eigenvalue(s) and corresponding eigenfunction(s) for criticality problems are most often computed with classical iterative methods such as the power iteration method. Convergence of the power iteration is determined by the dominance ratio, or the ratio of the second largest eigenvalue to the maximum eigenvalue.[1] Problems of practical interest often have dominance ratios large enough to make calculations of the maximum k-eigenvaluedifficult or intractable. Acceleration techniques may improve convergence, but they are often are not effective enough for the most difficult problems. This is because they rely on parameters which can not always be robustly estimated. Our purpose is to illustrate the use of an efficient Krylov subspace iterative method whose convergence is not limited by high dominance ratios for the calculation of the dominant eigenvalues for deterministic S_N transport methods. This method is the Implicitly Restarted Arnoldi Method (IRAM).[2] The IRAM is designed to be a robust and stable algorithm, with only the dimension of the Krylov subspace (the number of working vectors) as a free parameter. We suggest that the IRAM could improve the computational efficiency for criticality problems – those with high dominance ratios, in particular – relative to power iteration-based

To our knowledge, the work reported in this paper represents the first application of a Krylov subspace method for k-eigenvaluecalculations for large scale, three dimensional *transport* applications. However, a paper has recently appeared that describes the application of the IRAM to k-eigenvaluecalculations based on two-group diffusion theory.[3] Other alternatives to simple power iteration, most often for diffusion models, have also been investigated in the past. For example, a good deal of work appears in the Russian literature indicating that approaches such as inverse and shifted inverse iteration have been applied to eigenvalue computations.[4] Subspace iteration (which is similar in spirit to Krylov subspace methods) and variational acceleration of that method, has been recently investigated in the context of a diffusion model.[5] Another example is found in a recent paper describing inverse iteration for k-eigenvaluecalculations in one dimensional transport problems.[6]

We will describe how the IRAM can be easily implemented in an existing deterministic transport code using the freely available ARPACK software.[7] Because the IRAM solver is "wrapped around" the power iteration, the existing implementation does not need to be significantly altered. The IRAM is designed to be a robust and stable algorithm, with only the dimension of the Krylov subspace (the number of working vectors) as a free parameter. We have found that a Krylov subspace dimension of order five or ten seems to be optimum for our applications. The method can be applied equally well to both symmetric and nonsymmetric problems, so that the implementation does not need to be altered in order to ensure a symmetric transport operator.

Our implementation is in the AttilaV2 transport code,[8] a linear discontinuous finite element spatial discretization of the S_N equations on unstructured meshes, described in Sec. 2.1. This is followed by a discussion of the solution methods we consider for the k-eigenvalueproblem. This is followed by a discussion of the solution methods we consider for the k-eigenvalueproblem. This includes power iteration, discussed in Sec. 2.2, followed by qualitative introduction to the IRAM in Sec. 2.4. Numerical experiments are presented in Sec. 3 that illustrate the efficiency of the IRAM due to its high convergence rate. The first set of numerical results are presented in Sec. 3.1. Unaccelerated power iteration is compared to the IRAM for problems specifically constructed to have increasingly high dominance ratios. The next set of results, shown in Sec. 3.2, are intended to demonstrate the performance of the IRAM on a more realistic reactor problem.

We will show that any extra computational work or storage associated with the IRAM implementation is acceptable given the improvement in performance. This is certainly true when a problem does not have energy upscatter. However, in the case of upscatter an inner iteration energy is needed that detracts from the relative efficiency of the method compared to power iteration, which does not require such an inner iteration. This is a significant observation that was not considered in the diffusion model of Ref. 3. An additional interesting feature of the method is that the eigenvectors that correspond to several of the largest eigenvalues can also be calculated efficiently with little additional computational expense and no further code modification.

2 k-EIGENVALUE CALCULATIONS WITH S$_N$ TRANSPORT

We will start this section by describing the multigroup in energy and linear discontinuous finite element method (DFEM) spatial discretization of the S$_N$ equations on tetrahedral meshes that we consider in this paper. This is followed by a discussion of the power iteration method. A brief outline of the IRAM is then presented. The section finishes with a discussion of special considerations associated with implementing the Krylov iterative method in an existing S$_N$ transport code

2.1 Discrete S$_N$ Equations on Tetrahedral Meshes

We use standard notation.[1] Given an angular quadrature set with N specified nodes and weights $\{\hat{\Omega}_m, w_m\}$ and anisotropic scattering of Legendre order L, the steady-state S$_N$ transport equation for energy group $g = 1, \dots, G$ in a three-dimensional domain $\boldsymbol{r} \in V$ with boundary $\boldsymbol{r}_b \in \partial V$, is

$$\Big(\hat{\Omega}_m \cdot \nabla + \sigma_{t,g}(\boldsymbol{r})\Big)\, \psi_{g,m}(\boldsymbol{r}) = \sum_{g'=1}^{G} \sum_{l=0}^{L} \sigma_{l,g' \to g}(\boldsymbol{r}) \sum_{n=-l}^{l} Y_{ln}(\hat{\Omega}_m)\, \phi_{l,g'}^{n}(\boldsymbol{r}) + \frac{1}{k}\, \chi_g(\boldsymbol{r}) \sum_{g'=1}^{G} \nu\sigma_{f,g'}(\boldsymbol{r})\, \phi_{0,g'}^{0}(\boldsymbol{r}), \quad (1a)$$

for $m = 1, \dots, N_A$. Here, $Y_{ln}(\hat{\Omega})$ are the normalized spherical harmonics functions and the scalar flux moments are given by

$$\phi_{l,g}^{n}(\boldsymbol{r}) = \sum_{m=1}^{N_A} w_m Y_{ln}(\hat{\Omega}_m) \psi_{g,m}(\boldsymbol{r}). \quad (1b)$$

This is an eigenproblem with eigenvalue k.

The linear DFEM discretization on tetrahedra is derived with a Galerkin variational formulation. For a given energy group g, the angular flux is expanded in a set of four linear basis functions L_j on a tetrahedron T_s (cell index s):

$$\psi_{g,m,s}(\boldsymbol{r}) = \sum_{j=1}^{4} \psi_{g,m,j,s} L_j(\boldsymbol{r}). \tag{2}$$

The weak form of the transport equation is then constructed for each of the functions $L_i, i = 1, \ldots, 4$ on cell T_s for angle m:

$$\begin{aligned} &\int_{\partial T_s} \left(\hat{\Omega}_m \cdot \hat{n}\right) \psi^b_{g,m}\, L_i\, dS - \int_{T_s} \psi_{g,m,s}(\boldsymbol{r}) \left(\hat{\Omega}_m \cdot \nabla L_i\right) dV + \sigma_{t,g,s} \int_{T_s} \psi_{g,m,s}(\boldsymbol{r})\, L_i\, dV \\ &\quad = \sum_{g'=1}^{G} \sum_{l=0}^{L} \sigma_{l,g'\to g,s} Y_{ln}(\hat{\Omega}_m) \int_{T_s} \phi^n_{l,g'}(\boldsymbol{r})\, L_i\, dV + \frac{1}{k}\, \chi_{g,s} \sum_{g'=1}^{G} \nu\sigma_{f,g',s} \int_{T_s} \phi^0_{0,g'}(\boldsymbol{r})\, L_i\, dV. \end{aligned} \tag{3a}$$

This gives four equations for the four unknowns $\psi_{m,j,s}$ on every cell s for every angle $\hat{\Omega}_m$. Cross sections and other parameters are constant in a cell. The fluxes on the cell interfaces, $\psi^b_{g,m}$, are defined to make the approximation discontinuous as follows. For a face j on the boundary ∂T_s of cell s, whose outward normal is $\hat{n}_j$, we let

$$\left(\hat{\Omega}_m \cdot \hat{n}_j\right)\psi^b_{g,m} = \begin{cases} \left(\hat{\Omega}_m \cdot \hat{n}_j\right)\psi_{g,m,i(j),s}, & \hat{\Omega}_m \cdot \hat{n}_j > 0, \quad \hat{n}_j \text{ in } V \\ \left(\hat{\Omega}_m \cdot \hat{n}_j\right)\psi_{g,m,i(j),p}, & \hat{\Omega}_m \cdot \hat{n}_j < 0, \quad \hat{n}_j \text{ in } V\backslash\partial V \\ \left(\hat{\Omega}_m \cdot \hat{n}_j\right)\Gamma(\hat{\Omega}_m), & \hat{\Omega}_m \cdot \hat{n}_j < 0, \quad \hat{n}_j \text{ on } \partial V \end{cases} \tag{3b}$$

where p is the cell that shares face j with cell s. The subscript $i(j)$ denotes three vertices i on a face j of a given cell.

We will consider either vacuum boundary conditions, $\Gamma(\hat{\Omega}_m) = 0$, or specular reflection boundary conditions, where the reflected image of $\hat{\Omega}_m$, $\hat{\Omega}_{m'}$, is is defined by $\hat{\Omega}_{m'} = \hat{\Omega}_m - 2\,\hat{n}(\hat{\Omega}_m \cdot \hat{n})$. This determines an m' for $\hat{\Omega}_m$ and $\hat{n} = \hat{n}_j$ such that we can set $\Gamma(\hat{\Omega}_m) = \psi_{g,m',i(j),s}$. Reflective boundary faces are aligned parallel to the x, y or z coordinate axes such that most standard quadrature will contain reflected pairs $\hat{\Omega}_m$ and $\hat{\Omega}_{m'}$ that satisfy the definition of specular reflection.

In our implementation the integrals in Eqs. 3 are evaluated analytically. Note that we use a fully lumped version of Eqs. 3. Describing it goes beyond the scope of this work, but suffice it to say that this lumping preserves the diffusion limit in thick, diffusive regimes (see 9).

2.2 Power Iteration

We will now describe how the eigenvalue problem is solved with the power iteration method. Start by first writing the discretized S_N equations, Eqs. 3, in operator notation

$$\boldsymbol{L}\psi = \boldsymbol{MSD}\psi + \frac{1}{k}\boldsymbol{FD}\psi. \tag{4}$$

Detailed meanings of the operators can be deduced by comparing Eq. 4 with Eqs. 3. Briefly however, $\boldsymbol{L}$ is the transport operator, $\boldsymbol{S}$ is the scattering operator and $\boldsymbol{F}$ is the fission operator. The operators $\boldsymbol{M}$ and $\boldsymbol{D}$ represent the "moment-to-discrete" and "discrete-to-moment" operators, respectively, in the nomenclature of Ref. 10. They convert a vector of scalar flux moments to angular fluxes and vice versa, respectively. We will ignore boundary conditions to facilitate this discussion, without loss of generality. Rearranging Eq. 4, applying $\boldsymbol{D}$ to both sides, and introducing iteration index ℓ, we have

$$\phi^{\ell+1} = \frac{1}{k_\ell}\boldsymbol{D}\boldsymbol{H}^{-1}\boldsymbol{F}\phi^\ell, \tag{5}$$

where $\boldsymbol{H} = \boldsymbol{L} - \boldsymbol{M}\boldsymbol{S}\boldsymbol{D}$. The vector ϕ represents all the scalar flux moments contained in Eqs. 1 and $\phi^\ell = \boldsymbol{D}\psi^\ell$.

Power iteration computes the largest eigenvalue k by iterating Eq. 5 until the expression

$$k_{\ell+1} = k_\ell \frac{\|\phi^{\ell+1}\|}{\|\phi^\ell\|}, \tag{6}$$

converges to within some tolerance, where $\|\phi\|$ represents some discrete norm of ϕ over all cells in the problem. In many implementations the norm takes into account only the isotropic scalar flux moment component of ϕ, ϕ_0^0, although there is no mathematical justification for this. Furthermore, the eigenvalue estimate is often updated based on the total fission rate in the problem.[1] That is, the norm in Eq. 6 is taken to be

$$\|\phi\| \equiv \|\phi\|_F = \sum_{g=1}^{G}\sum_{s}\nu\sigma_{f,g,s}\phi_{0,g,s}^0, \tag{7}$$

where the sum over s is for all cells in the problem.

An alternative to Eq. 6 is to use the Rayleigh quotient to update the eigenvalue estimate as follows:

$$\begin{aligned} k_{\ell+1} &= \frac{(\boldsymbol{A}\phi^\ell, \phi^\ell)}{(\phi^\ell, \phi^\ell)} && (8)\\ &= k_\ell \frac{(\phi^{\ell+1}, \phi^\ell)}{(\phi^\ell, \phi^\ell)}, && (9)\end{aligned}$$

where $(\cdot,\cdot)$ is a discrete inner product over all cells, and $\boldsymbol{A} = \boldsymbol{D}\boldsymbol{H}^{-1}\boldsymbol{F}$. We have observed that using the Rayleigh quotient rather than Eq. 6 for the eigenvalue estimate can sometimes improve the efficiency of the power iteration method by providing a better estimate of the eigenvalue earlier in the iterative process.[11] It appears that the Rayleigh quotient estimate can, in some cases, lead to faster convergence of the inner iterations and better overall efficiency.

It can be shown that power iteration will converge to the largest eigenvalue in magnitude k_1 (the dominant eigenvalue) and that the convergence of this iteration is determined by the dominance ratio $\delta = k_2/k_1$ where

$k_2 \leq k_1$ is the next largest eigenvalue in magnitude.[11] The closer this ratio is to one the more slowly power iteration converges. In the general case, it is possible that power iteration will not converge. This can happen, for instance, if the eigenvalue with largest magnitude is complex and the operator is real and nonsymmetric and the initial guess is real. In the case of monoenergetic transport it has been shown that the k-eigenvalues of the S_N approximation are always real, even for anisotropic scattering.[12] However, we do not know if this is also true for the multigroup approximation so we cannot state whether or not there exist multigroup problems for which the power iteration method might not converge.

2.3 Acceleration of Power Iteration

Power iteration works with two vectors, or iterates, $\phi^{\ell+1}$ and ϕ^{ℓ}. Convergence can be improved by storing additional vectors and making use of previous information. Chebyshev iteration, successive over-relaxation (SOR), and other classical iterative techniques are commonly used to accelerate k-eigenvaluepower iterations in this way.[13, 14] Such methods use a linear combination of several of the previous iterates to update the scalar fluxes. Under certain circumstances it is possible to use knowledge of the spectrum of the operator to find the optimal linear combination that minimizes the convergence rate. In practice the spectrum is not known in advance and estimates of the spectrum are made to compute the necessary parameters. The result is a less-than-optimal method. When such methods are applied to general transport problems, such as those that include anisotropic scattering or energy dependence, the tranpsort operator is nonsymmetric. In this case, the spectrum could extend into the complex plane and methods for computing optimal estimates of the required acceleration parameters becomes significantly more difficult.[15–17] This consideration is often ignored in practice and acceleration methods are implemented based on the assumption that the operator is symmetric and positive definite. This creates the potential for degrading or destroying the effectiveness of the acceleration. The heuristics, approximations, and code logic needed to make adaptive estimates of the required parameters can also make these acceleration algorithms less than robust. The uncertain nature of such algorithms make it difficult to predict when or if a given acceleration method will be successful; while a particular approach may work well in some problems, it may work poorly in others and may even cause the iteration to diverge in others.[14]

2.4 The Implicitly Restarted Arnoldi Method

In the notation of Section 2.2, we are solving a standard eigenvalue problem of the form

$$\boldsymbol{D}\boldsymbol{H}^{-1}\boldsymbol{F}\phi = k\phi. \tag{10}$$

Our approach is to compute the dominant eigenvalue(s) k using the IRAM implementation found in the ARPACK software package.[7] The IRAM is a recently developed method for computing large scale eigenvalue problems. The ARPACK implementation is best suited for applications whose matrices are either sparse or not explicitly available; Only the "action" of the operator on a vector, supplied by the solver, has to be computed at every IRAM iteration. A reverse communication mechanism frees the user from any particular data structure formats. This simplifies implementation of the method in an existing transport code that uses the power iteration method because the IRAM solver can simply be "wrapped" around the power iteration coding. The desired number of extremal eigenvalues that are largest or smallest in magnitude, the maximum dimension of the Krylov subspace and the iterative convergence criteria are the only input supplied to the ARPACK software. Some overhead, in terms of both memory requirements and computation per iteration, depends on the size of the Krylov subspace and the number of requested eigenvalues. Currently we are treating the IRAM as a black box eigenvalue solver. In essence this means that the existing data structures for the scalar flux moment unknowns in the transport code are repeated. The ARPACK implementation is completely general in that it is not restricted to self-adjoint problems although the package does provide facilities that take advantage of symmetry leading to greater efficiency. Our spatially discretization of the discrete ordinates transport equation is never self-adjoint, even when using symmetrization techniques suggested in Refs. 18 and 19. We are therefore unable to take advantage of the symmetric IRAM implmentation in ARPACK.

We will now briefly describe the IRAM. First, consider the power iteration method, which is a simple way to compute the dominant eigenvalue and corresponding eigenvector – the fundamental mode – of $\boldsymbol{A}x = \lambda x$. (in our case $\boldsymbol{A} = \boldsymbol{D}\boldsymbol{H}^{-1}\boldsymbol{F}$ and $\lambda = k$). Only two vectors of storage are necessary and only the action of the operator $\boldsymbol{A}$ on a vector is required at every iteration. This can viewed as a Krylov subspace iteration with a subspace dimension of one. Now, imagine applying power iteration to several vectors, say p of them, simultaneously. The result would be that the p vectors would all converge to the dominant eigenvector. However, if the vectors were orthogonalized and normalized at every iteration they would instead converge to the p eigenvectors that correspond to the p eigenvalues largest in magnitude. This approach is called subspace iteration.

A natural question to ask is whether it is possible to store and combine several vectors in an attempt to take greater advantage of work that has already been performed. Power iteration operates on only the most recently computed vector, destroying any additional eigenvector information that might have been extracted from the previous iterations. The acceleration techniques mentioned previously in Section 2.3, do use information from the previous iterates, but often in a less than optimal way. However, suppose a starting vector for power iteration was expanded in terms of the eigenvectors of the operator. Then the expansion coefficients of the vector would evolve in a specific and identifiable way through the repeated application of the operator. A linear combination of the vector sequence generated during the course of the iterations, together with knowledge of this pattern of evolution, might then be constructed in such a way as to extract additional useful eigenvector information from the vector sequence that is optimal in a certain sense.

The Arnoldi method, which forms the basis of the IRAM, combines these two ideas. The IRAM is a Krylov subspace iterative method. The Krylov subspace of dimension m is

$$\mathcal{K}_m(\boldsymbol{A}, \phi_0) = \text{span}\{\phi_0, \boldsymbol{A}\phi_0, \ldots \boldsymbol{A}^{m-1}\phi_0\} \tag{11}$$

is constructed from the same vector sequence generated by power iteration applied to an initial starting vector ϕ_0. An approximate eigenvector-eigenvalue pair, or eigenpair, is found by projecting a vector $x \in \mathcal{K}_m(\boldsymbol{A}, \phi_0)$ onto the Krylov subspace using the Galerkin orthogonality condition $(w, \boldsymbol{A}x - \lambda x) = 0$ for all $w \in \mathcal{K}_m(\boldsymbol{A}, \phi_0)$. The approximate eigenpair (x, λ) is called a "Ritz pair". If the component of the actual eigenvector that is orthogonal to the subspace is small, then the Ritz pair will be a good approximation to the eigenpair of a nearby problem. The quality of this approximation is measured by the backward error.

Ritz pairs that satisfy the Galerkin orthogonality condition are computed efficiently using the Arnoldi method, which computes an orthogonal basis—the Arnoldi basis—for $\mathcal{K}_m(\boldsymbol{A}, \phi_0)$. At the same time, an (upper Hessenberg) $(m \times m)$ projection of $\boldsymbol{A}$ on the Krylov subspace is computed. The Ritz pairs are then easily found by the QR decomposition[20] of this matrix. An estimate of the (backward) error in the approximation, or Ritz estimate, is also available through the Arnoldi method. Typically, the quality of the approximation improves as the dimension m of the subspace increases. In exact arithmetic, they are equal after n iterations, where n is the order of $\boldsymbol{A}$.

The cost of maintaining orthogonality of the Arnoldi basis increases with the dimension of the Krylov subspace. The computations could become intractable if the size of the subspace needed for a good approximation to the eigenpair is too large. One way to address this is to fix the computational and storage requirements by restarting the Arnoldi method. Suppose that we are able to compute m steps of the Arnoldi method where m is chosen so that the cost of maintaining the orthogonality among the Arnoldi vectors (to machine precision) is small. Because we are interested in the s dominant eigenvalues of $\boldsymbol{A}$, consider constructing another set of Arnoldi vectors for $\mathcal{K}_m(\boldsymbol{A}, \hat{\phi}_0)$ such that the first s Arnoldi vectors of this new space span the same space as the Ritz vectors that correspond to the s dominant Ritz values of the original space $\mathcal{K}_m(\boldsymbol{A}, \phi_0)$. This is how the Arnoldi method is *restarted*, continuing until the s dominant eigenvalues emerge to the specified tolerance. Implicit restarting is an efficient and numerically stable way to restart the Arnoldi method or, equivalently, to compute $\hat{\phi}_0$, the new initial vector.[2,21] The restart is called implicit because of the connection with the implicitly shifted QR algorithm. See Ref. 7 for further details on an efficient implementation of implicit restarting. In particular, implicitly restarting an Arnoldi method is equivalent to an accelerated subspace iteration[22] (see Ref. 3 for a performance comparison of subspace iteration and the IRAM on a two-group diffusion problem).

In contrast to power iteration, which converges to the dominant eigenvalue k_1 with convergence rate

$|k_2|/|k_1|$, the IRAM converges at the rate

$$\frac{\max_{j=r+1,\ldots,n} |P(k_j)|}{\min_{j=1,\ldots,r} |P(k_j)|} \tag{12}$$

after every restart, where n again is the order of $\boldsymbol{A}$, r is the number of Arnoldi vectors remaining after an implicit restart and $P(x) = (x - x_1) \cdots (x - x_r)$. The zeros of the polynomial $P(x)$, $x_i, i = 1, \ldots, r$, and r, are selected so that (12) is as small as possible. Optimal choices do not, in general, exist and would require knowledge of the eigenvalues that we are attempting to approximate. If $r = 1$ and $x_1 = 0$, the power iteration convergence rate is recovered. If $r > 1$, and $x_i = 0$ for $i = 1 \ldots, r$, the convergence rate of subspace iteration is recovered. For both of these cases, r is chosen slightly larger than s (the desired number of eigenvalues), which is a well-known strategy in subspace iteration. The default choice in ARPACK is to choose the roots of $P(x)$ as the *unwanted* eigenvalues of the upper Hessenberg matrix computed during the Arnoldi method. These are the $m - r$ eigenvalues smallest in magnitude. This makes an excellent choice in practice and produces a convergence rate smaller than $|k_2|/|k_1|$. Moreover, this choice of $P(x)$ is mathematically equivalent to computing a new Krylov space with a starting vector that is a linear combination of the r dominant Ritz vectors. The value of r is chosen slightly larger than s in practice, which tends to decrease the ratio (12). More theoretical details can be found in Refs. 2, 21, and 22.

Finally, note that power iteration can actually be viewed as a Krylov method with a subspace of dimension one. Similarly, classical techniques used to accelerate power iteration can be viewed as a Krylov method of a small dimension that is equal to the number of working vectors.[23] As we mentioned in Sec. 2.3, the parameters for computing appropriate linear combination of these vectors have to be estimated or approximated in practice, meaning that the methods are non-optimal and may limit their effectiveness. In contrast, the IRAM automatically and efficiently selects the Ritz pair(s) without any a-priori information or parameter estimation.

2.5 Special Considerations

Let us focus on the energy dependence of the operator $\boldsymbol{A} = \boldsymbol{D}\left(\boldsymbol{L} - \boldsymbol{MSD}\right)^{-1}\boldsymbol{F}$ for the moment. The operators $\boldsymbol{L}$, $\boldsymbol{M}$ and $\boldsymbol{D}$ are block diagonal, where each block corresponds to the discretized equations for a particular energy group. We can split the scattering operator into its lower triangular part, corresponding to downscatter and within-group scattering, and its strictly upper triangular part, corresponding to upscatter, such that $\boldsymbol{S} = \boldsymbol{S}_L + \boldsymbol{S}_U$. Equation 4 becomes

$$\left(\boldsymbol{L} - \boldsymbol{MS}_L\boldsymbol{D}\right)\psi = \left(\boldsymbol{MS}_U\boldsymbol{D} + \frac{1}{k}\boldsymbol{FD}\right)\psi. \tag{13}$$

Upon rearranging, operating from the left with $\boldsymbol{D}$, and introducing the power iteration index ℓ, this becomes

$$\phi^{\ell+1} = \boldsymbol{D}\left(\boldsymbol{L} - \boldsymbol{M}\boldsymbol{S}_L\boldsymbol{D}\right)^{-1}\left(\boldsymbol{M}\boldsymbol{S}_U + \frac{1}{k}\boldsymbol{F}\right)\phi^{\ell} \tag{14}$$

If a problem consists of downscatter only, then the operator $\boldsymbol{S}_U$ is zero and $\boldsymbol{S} = \boldsymbol{S}_L$ is block lower triangular. In that case, the inverse $\boldsymbol{H}^{-1} = \left(\boldsymbol{L} - \boldsymbol{M}\boldsymbol{S}_L\boldsymbol{D}\right)^{-1}$ can be computed by block forward substitution, from high energy to low energy in sequence. Each step of forward substitution involves an inner iteration for a single energy group, calculated either with traditional source iteration or with a Krylov iterative method as described in Ref. 24. Thus $\boldsymbol{H}^{-1}$ represents an approximate inverse that is computed to within some specified inner iteration convergence tolerance.

If upscatter is present, then the action of $\boldsymbol{H}^{-1} = \left(\boldsymbol{L} - \boldsymbol{M}\boldsymbol{S}\boldsymbol{D}\right)^{-1}$ can be viewed as being "embedded" in the power iteration method because power iteration works with a single vector of scalar fluxes. This can be made clear if we suppose that we are considering a fixed-source problem and not an eigenvalue problem, with some source term Q replacing the fission source $\frac{1}{k}\boldsymbol{F}$. In that case, Eq. 14 is an outer Gauss-Seidel iteration for the scalar flux moments. It is in this sense, then, that the approximation to the inverse operator $\boldsymbol{H}^{-1}$ is calculated "implicitly" - via a Gauss-Seidel iteration - in the course of the power iterations.

We are still investigating how we might do something similar with the IRAM. At this time, however, if we want to compute the eigenvalues for problems involving upscatter with ARPACK, we have to compute the inverse $\boldsymbol{H}^{-1}$ with some other iterative method at every IRAM iteration. We can do that either with a Krylov method or with the block Gauss-Seidel iteration that is part of the power iteration coding. The algorithm now consists of three nested levels of iteration, the outermost level being the IRAM iteration and the lowest level being the within group inner iterations and an intermediate level for computing $\boldsymbol{H}^{-1}$ (which is typically thought of as the outer iteration). Unless a very efficient acceleration or preconditioning technique could be developed, this approach is likely to be less efficient compared to power iteration in many cases because One possibility would be to precondition a Krylov iterative method or accelerate the block Gauss-Seidel iteration with the upscatter acceleration method of Ref. 25. In this paper, we will not consider problems with upscatter although we plan to address this issue in the future.

Another important aspect of power iteration that has a significant impact on its efficiency is that the current scalar flux moments from the most recently computed scalar fluxes can be used as initial guesses for the next set of inner iterations. This is why the eigenvalue problem is implemented as shown in Eq. 5. Scaling the eigenvector with the most recent eigenvalue estimate k_ℓ not only prevents overflow or underflow but also enables us to use the solution from the previous iteration as an initial guess for computing $\boldsymbol{H}^{-1}$. This aspect of the power iteration algorithm significantly reduces the overall computational expense.

Although similar good initial guesses are not available for subsequent inner iterations in the IRAM iterations, there is an effective approach for relaxing the inner iteration convergence tolerance during the course of the IRAM iterations to improve efficiency. We called this approach the BFG (Bouras, Frayssé and Giraud) strategy when we originally used it for Krylov iterations preconditioned by an inner DSA conjugate gradient iteration.[26] The authors of Refs. 27, 28, and 29 observed that that Krylov subspaces are built in sequence, with every subsequent vector depending on the previous vectors. Then, loosely speaking, if the action of the operator is computed iteratively, which can alternatively be looked upon as an approximate or "inexact" matrix-vector product, it is important to compute the vector operations (inner iterations) to a strict convergence tolerance in the early outer iterations. The convergence criteria can then be relaxed as the outer iteration proceeds. By choosing the inner tolerance to be inversely proportional to the outer residual the computational work in the inner iterations is reduced without affecting either the outer convergence rate or the solution. If we wish to increase the inner convergence tolerance more slowly we can take it to be proportional to some inverse fractional power of the outer residual as well, although we have not found that to be necessary. However, some very recent work has uncovered a theoretical justification for such an approach, for both linear systems solution and eigenproblems.[30] In that work, a constant of proportionality for the inverse relationship of the inner iteration tolerance and the outer residual is derived. A proportionality constant based on theoretical arguments is compared to an ad hoc approach of simply setting it to unity and in many cases, but not all, there was only a little difference between the two. We selected the constant of proportionality to be 0.1. In the problems we have experimented with to date, including solutions to linear systems as well as the eigenvalue problems discussed in this paper, we have not encountered any difficulties with this strategy.

In any case, the inner iterations are usually accelerated or preconditioned with diffusion synthetic acceleration (DSA), although in some cases it may be more efficient to compute the inner iterations without DSA, particularly if inner Krylov iterations are used. Boundary conditions are implemented naturally in terms of the angular fluxes during the course of the inner iterations. We will find that there are some problems where, because of the efficiency of the power iteration implementation, we can use a few power iterations to initialize the IRAM and improve its convergence.

Finally, note that the IRAM is not the only existing method that can be used to improve upon power iteration.

3 NUMERICAL RESULTS

In this section we compare the IRAM to the power iteration method already implemented in the three-dimensional, tetrahedral mesh, transport code AttilaV2. We will present actual measurements of the computational cost for representative problems. We use CGS units and a triangular S_4 Chebyshev-Legendre quadrature for all the results reported here. Calculations are started with random initial vectors. In cases where power iteration is used to initialize the IRAM, the power iteration method is initialized with a random vector and the starting vector for the IRAM is initialized with the result of a few power iterations. Because of the unpredictable nature of the classical acceleration methods for general problems discussed in Section 2.3, we only consider unaccelerated power iteration for comparison.

The question of how to measure convergence is a significant issue for iterative solution methods. The ARPACK solvers use the backward error,

$$\|\boldsymbol{A}x - \theta x\|_2 \leq |\theta|\epsilon \tag{15}$$

to determine convergence, where x and θ are the current approximate eigenpair, and the tolerance ϵ is specified. Rather than calculate the residual explicitly, ARPACK uses the Ritz estimate, an indirect measure of the error in the associated eigenpair, that is readily available through the Arnoldi factorization.[7] The norm of the residual at step ℓ is

$$\begin{aligned} \|r^\ell\|_2 &= \|\boldsymbol{A}\phi^\ell - k_\ell\phi^\ell\|_2 \\ &= \|k_\ell\phi^{\ell+1} - k_\ell\phi^\ell\|_2 \\ &= |k_\ell|\|\phi^{\ell+1} - \phi^\ell\|_2, \end{aligned} \tag{16}$$

so we can use $\|\phi^{\ell+1} - \phi^\ell\|_2 \leq \epsilon$ as the convergence criteria for the power iteration method. This is in rough agreement with the convergence criteria in ARPACK, Eq. 15. We feel that this makes a comparison of the computational expense as fair as can be expected.

The outer iteration convergence criterion for all the results is $\epsilon = 10^{-4}$. The power iteration results are computed using traditional source iteration accelerated with the partially consistent simplified WLA DSA method[26, 31] using the previous outer iteration solutions for the initial guess. The inner iteration convergence criterion is proportional to the outer iteration residual $\|r^\ell\|$, that is, we set $\epsilon = \min(10^{-2}, 10\|r^\ell\|)$ at outer iteration ℓ. This is a commonly used approach in Newton-type methods (like power iteration) that reduces the total amount of work, although it may sometimes delay the outer iteration convergence, without affecting the accuracy of the solution. In this case having a good initial guess available for each within-group inner iteration, together with an effective DSA scheme, minimizes any deleterious effects on convergence.

3.1 Cylindrical Mesh Problem

The first set of results is for a series of problems with dominance ratios near and approaching one. The problems are constructed by altering the symmetry of a cylindrical mesh, illustrated in Fig. 1. The cylinder is 3.5 cm in radius and 9 cm long. It consists of a 5 cm layer of B^{10} absorber sandwiched between 1 cm thick water layers and 1 cm layers of highly enriched uranium (HEU). Vacuum boundary conditions are used on all the faces and there 13,500 cells in the mesh. Five 0.1 cm thick regions at the left of the central absorbing

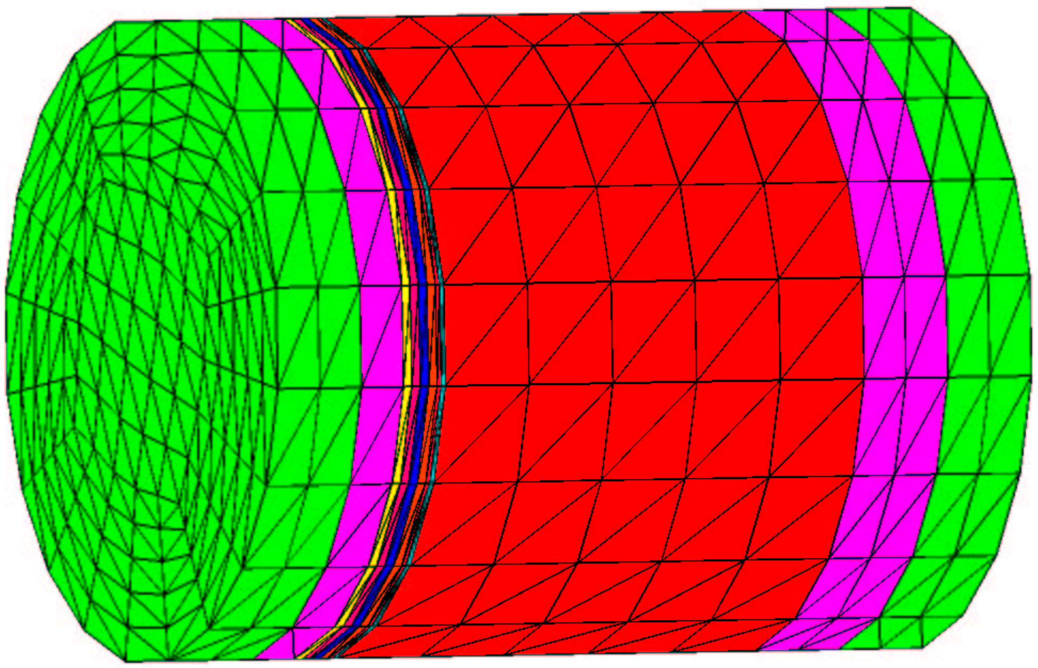

Figure 1: The cylindrical mesh problem. Note the thin disks in the left half of the mesh that are used to alter the symmetry of the problem.

region can be seen in Fig. 1. The configuration is symmetric if all five regions are filled with water. The dominance ratio of this symmetric configuration is very close to one, a situation in which power iteration can be expected to converge very slowly. By successively substituting boron for water in the thin regions starting with the region nearest the absorber region we destroy the symmetry and reduce the dominance ratio in the problem. While the actual value of the eigenvalue does not change significantly with the departure from symmetry, the dominance ratio and fundamental eigenvector do.

Five energy group cross sections and fission data collapsed from Hansen and Roach sixteen group cross section data sets[32, 33] were used. The material composition data with material IDs from the cross section library are shown in Table 1. The energy groups are collapsed as follows:

Group	Library Groups
1	1 – 2
2	3 – 4
3	5 – 8
4	9 – 12
5	13 – 16

The results for this sequence of problems are shown in Figs. 2, 3, and 4, and in Tables 2 and 3. The inner, within-group iterations for these problems were computed using restarted, flexible-GMRES, FGMRES(10).

Table 1: Cross section data for the cylindrical mesh problem. Density is in g/cm^3.

Material	Density	ID	Mass Fraction
HEU	19.0	u252e1 u282e1	0.95 0.05
Water	1.0	hDE o	0.667 0.333
Boron	10.0	b10	1.0

Figure 2 simply verifies the dependence of power iteration convergence on dominance ratio. Note in the

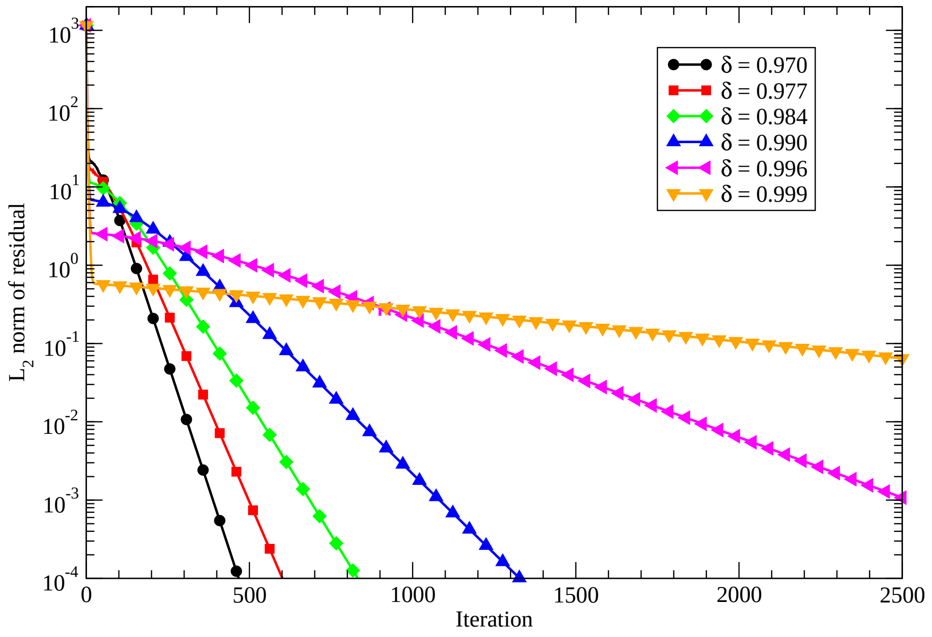

Figure 2: Convergence curves for power iteration illustrating the decreasing convergence rates for increasing dominance ratio, δ.

figure that the two problems with largest dominance ratios did not converge to 10^{-4} within 2500 iterations. Figure 3 shows the how the computational expense (floating point operation (FLOP) counts measured on a single dedicated 250 MHz SGI Origin 2000 CPU) corresponds to the slower convergence and shows that most of the computation is devoted to the within-group inner iterations. Thus it is possible to gauge the relative performance of the two methods by simply comparing inner iteration counts. Figure 4 illustrates the one to two orders of magnitude savings in computational cost with the IRAM relative to unaccelerated power iteration. The different curves in the figure are for several combinations of the number of eigenvalues sought, `nev` = 0 or 1, the maximum dimension of the Krylov subspace, `ncv` = 5 or 10, and the number of power iterations taken in initializing the IRAM iterations, `init` = 0 or 5. These values were chosen for illustration only. The figure shows that the resulting computational expense is not very sensitive to either

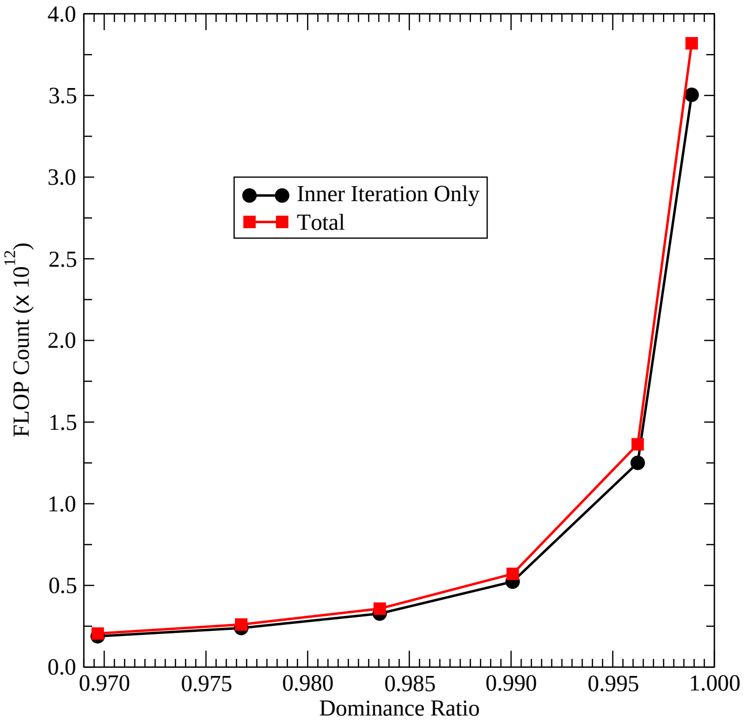

Figure 3: FLOP (floating point operation) counts for power iteration.

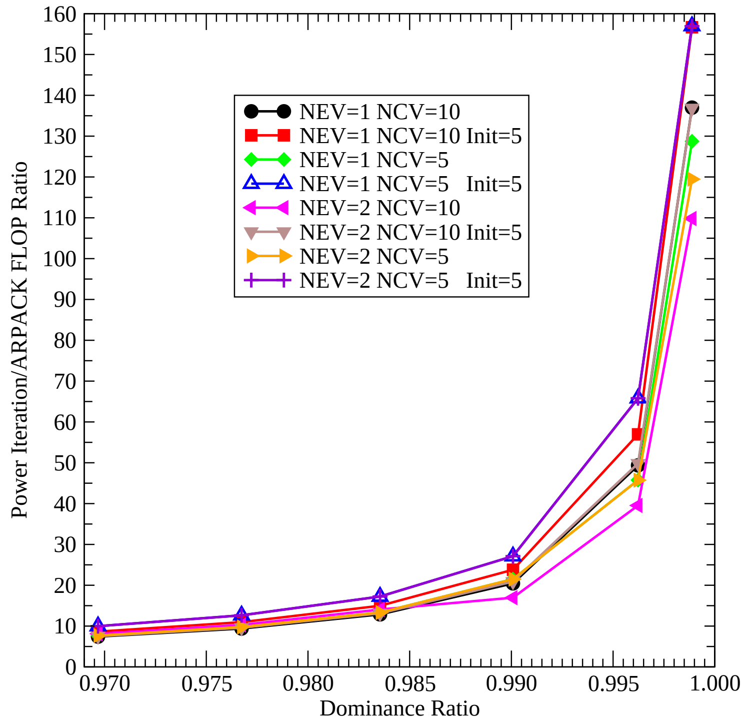

Figure 4: Relative computational expense of ARPACK compared to power iteration.

`nev` or `ncv` while the initialization with a few power iterations is certainly worth the cost. For efficiency, the initialization steps involved a single inner iteration for each group for each power iteration and this was enough, apparently, to achieve a good initial guess. We did not plot the IRAM convergence curves in Fig. 2 because the convergence rate of the IRAM was so much faster than the power iteration method and would not be distinguishable on the scale of that the figure. However, for completeness, in Tables 2 and 3 we list the number of outer and total number of inner iterations, respectively. The most interesting observation is that convergence of the IRAM is largely insensitive to the dominance ratio. The computational expense depends more on the size of the Krylov subspace than on dominance ratio, as seen in the inner iteration data. We also see that asking for a second eigenvector does not significantly increase the number of outer or inner iterations. Finally, the reduction in the number of outer iterations achieved by initializing the IRAM with power iteration results in the lowest overall amount of computational work.

Keep in mind that it is the eigenvector and not the eigenvalue that we used to determine convergence of both the power iteration algorithm and the IRAM. The last line of entries in Table 2 shows the number of outer power iterations needed to converge the eigenvalue to an absolute error of 10^{-5}. This illustrates that the eigenvalue typically converges much more quickly than does the eigenvector for the power iteration method, a property the IRAM does not share. In the case of a symmetric operator, for instance, it is well known that convergence is quadratic in the dominance ratio for the eigenvalue and linear for the eigenvector.[34]

Table 2: Number of outer iterations for the IRAM on the cylindrical mesh problem. Power iteration results shown for comparison. The "k only" entry shows the number iterations for the power iteration method to converge the eigenvalue to an absolute error of 10^{-5}.

Method			Dominance Ratio, δ					
`nev`	`ncv`	`init`	0.970	0.977	0.984	0.990	0.996	0.999
1	10	0	20	20	20	20	20	20
1	10	5	15	15	15	15	15	15
1	5	0	20	20	20	20	23	23
1	5	5	14	14	14	14	14	17
2	10	0	18	18	18	25	26	26
2	10	5	18	18	18	18	18	18
2	5	0	20	20	20	20	23	25
2	5	5	14	14	14	14	14	17
Power Iteration			468	602	833	1,329	3,170	8,824
k only			229	287	388	586	1,206	1,739

Table 3: Total inner iterations for the IRAM on the cylindrical mesh problem. Power iteration results shown for comparison.

Method			Dominance Ratio, δ					
`nev`	`ncv`	`init`	0.970	0.977	0.984	0.990	0.996	0.999
1	10	0	460	460	458	457	447	447
1	10	5	392	390	389	389	385	390
1	5	0	440	436	438	428	472	463
1	5	5	327	327	327	332	322	372
2	10	0	424	424	422	544	550	549
2	10	5	446	441	440	438	436	443
2	5	0	440	436	438	428	472	497
2	5	5	327	327	327	332	322	373
Power Iteration			2,503	3,124	4,218	6,691	15,892	44,156
k only			1,308	1,549	1,993	2,976	6,077	8,747

3.2 MOX Fuel Assembly Problem

The second numerical example is for a three dimensional MOX fuel assembly benchmark problem (C5G7 MOX) developed by the Expert Group on 3-D Radiation Transport Benchmarks. The interested reader can consult Appendix C in Ref. 35 for detailed specifications of this problem. Briefly, however, the problem consists of four 17×17 pin (with cladding) fuel assemblies containing $U0_2$ and 4.3% MOX, 7.0% MOX and 8.7% MOX fuels in various configurations, surrounded by moderator (water). Guide tubes and fission chambers are also present in the assemblies. The overall dimensions are $64.26\text{cm} \times 64.26\text{cm} \times 214.20\text{cm}$ in the x, y, and z dimensions. Reflective boundary conditions are specified on the $x = 0$, $y = 0$, and $z = 0$ faces. Vacuum boundary conditions are specified on the other three faces. The tetrahedral grid constructed for this problem consists of 954,527 cells with 169,745 vertices. Seven group, transport-corrected, isotropic scattering cross section data are given. The problem consists of nearly 28 million degrees of freedom, a substantial size for a serial implementation. This problem is intended to show the relative merits of the two methods for large, realistic transport problems. However, the upscatter portion of the scattering matrix is set to zero in order to compare power iteration calculations to the IRAM calculations. The inner, within-group iterations were computed using BiCGStab.

The dominance ratio for this problem was approximately 0.985 so that the power iteration method converged slowly, taking 6,144 inner iterations over the course of 860 outer iterations. We saw in the first set of results that the total number of inner iterations gives a very good idea of the relative performance of the methods. Thus the total number of outer and inner iterations for the two methods are shown in Table 4.

The power iteration calculation took about 150 hours of wall clock time on a dedicated 1000 MHz 64-bit Compaq Alpha EV6.8CB (21264C) CPU with an internal Alpha FPU and 8 GB RAM. For comparison, the ARPACK calculations ran from about 23 to 34 hours of wall clock time. As in the previous set of results, we

Table 4: Iteration counts for the three dimensional MOX fuel assembly benchmark problem. The "k only" entry shows the number iterations for the power iteration method to converge the eigenvalue to an absolute error of 10^{-5}.

Method			Iterations	
`nev`	`ncv`	`init`	Outer	Inner
1	10	0	25	1,109
1	10	5	25	1,050
1	5	0	38	1,404
1	5	5	32	1,132
2	10	0	34	1,396
2	10	5	33	1,303
2	5	0	44	1,578
2	5	5	38	1,299
Power Iteration			860	6,144
k only			296	2,196

see that initializing the IRAM with power iteration (one inner iteration per power iteration initialization step for each group) can make a difference. We found that taking more than five initialization iterations did not change the number of outer iterations and had only a small affect on the number of inner iterations for this problem. Overall, the IRAM is roughly five times faster than power iteration for this problem. Accelerating the power iteration method with a Chebyshev, SOR or some other classical acceleration technique, could conceivably affect this conclusion. Note that we tried simple SOR acceleration on this problem but could not find a robust sequence of relaxation parameters that would accelerate convergence without causing the iteration to diverge. Once again, convergence of the eigenvalue was much faster than the eigenvector for power iteration.

We can make some general observations of a qualitative nature regarding the choice of `nev`, `ncv` and `init`. Convergence of the IRAM is more sensitive to number of eigenvalues requested, `nev`, than it is to the maximum Krylov subspace dimension, `ncv`. We also find that overall computational cost is more sensitive to a good initial starting vector than is the convergence of the outer iteration because of the reduction in the outer iteration count. Finally, our experience suggests that the IRAM is insensitive to dominance ratio.

4 CONCLUSIONS

The Implicitly Restarted Arnoldi Method as available in the software package ARPACK was implemented in our the three dimensional, unstructured tetrahedral mesh, linear discontinuous discrete ordinates transport code AttilaV2. We were able to easily implement the method using the existing outer and inner iteration coding and other available computational machinery. It would most likely be just as easy to implement in any other code.

Our numerical experiments, although certainly not exhaustive, show that the method is robust and extremely efficient for several difficult representative problems, when compared to unaccelerated power iteration. Any incerease in work per iteration associated with the eigenvalue solver is vastly outweighed by the improvement in convergence rate. The small additional memory requirement is tolerable, especially in light of the improved performance.

The ARPACK implementation of IRAM has been extended to parallel platforms[36] so the method can be just as easily implemented in existing parallel transport codes as it can in serial codes. The results and performance reported here, and hence our conclusions, could be different on parallel platforms, although we do not have any reason at this time to expect that the IRAM will perform any better or worse in parallel.

The IRAM is obviously not limited to the discretized S_N transport equation used in the numerical results presented here. We believe that the IRAM could perform just as well, relative to power iteration, for other methods. Improvements in efficiency could be obtained if the IRAM were implemented directly into the transport source code in an "in-line" fashion and optimized to use the existing data structures and storage that has already been allocated. Perhaps existing acceleration methods for multigroup fission problems, like Chebyshev or coarse-mesh rebalance, could be used to further improve the overall efficiency of the IRAM-based k-eigenvaluecalculations. Other types of iterative eigenvalue methods, such as shifted-inverse iteration, might also be combined with the IRAM to improve the efficiency. We have shown one example already, having used power iteration to initialize the IRAM and speed up convergence. A significant drawback of the method as it is currently implemented is the inability to treat problems with energy upscatter efficiently. We plan to address this in the future. Nonetheless, the potential of the IRAM for use in k-eigenvaluecalculations is clear.

Acknowledgments

The authors would like to thank Michele Benzi of Emory University for his very helpful discussions on this and related topics and for helping to initiate this collaboration. The work of the first four authors was performed under the auspices of the United States Department of Energy and the National Nuclear Security Agency at the Los Alamos National Laboratory. The work of the fifth author (Lehoucq) was performed at the Sandia National Laboratories, a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy under contract DE-AC04-94AL85000.

References

1. E. E. Lewis and W. F. Miller, **Computational Methods of Neutron Transport**. Wiley & Sons: New York (1984).

2. D. C. Sorensen, "Implicit Application of Polynomial Filters in a k–step Arnoldi Method," *SIAM J. Matrix Anal. Appl.*, **13**, n. 1, pp. 357–385 (1992).

3. G. Verdú, R. Miró, D. Ginestar, and V. Vidal, "The Implicit Restarted Arnoldi Method, An Efficient Alternative to Solve the Neutron Diffusion Equation," *Annals of Nuclear Energy*, **26**, pp. 579–593 (2002).

4. G. I. Marchuk and V. I. Lebedev, **Numerical Method in the Theory of Neutron Transport**. Harwood Academic Publishers: Chur, Switzerland, Revised Second Edition (1986). Translated and edited by O. Germogenova.

5. V. Vidal, G. Verdú, D. Ginestar, and J. L. Munõz-Cobo, "Variational Acceleration for Subspace Iteration Method. Application to Nuclear Power Reactors," *International Journal for Numerical Methods in Engineering*, **41**, pp. 391–407 (1998).

6. E. J. Allen and R. M. Berry, "The Inverse Power Method for Calculation of Multiplication Factors," *Annals of Nuclear Energy*, **29**, pp. 929–935 (2002).

7. R. B. Lehoucq, D. C. Sorensen, and C. Yang, **ARPACK User's Guide**. SIAM: Philadelphia (1998).

8. T. A. Wareing, J. M. McGhee, and J. E. Morel, "ATTILA: A Three-Dimensional, Unstructured Tetrahedral Mesh Discrete Ordinates Transport Code," *Transactions of the American Nuclear Society*, **75**, pp. 146–147 (1996).

9. M. L. Adams, "Discontinuous Finite Element Methods in Thick Diffusive Problems," *Nucl. Sci. Engr.*, **137**, pp. 298–333 (2001).

10. J. E. Morel, "A Hybrid Collocation-Galerkin-S_n Method for Solving the Boltzmann Transport Equation," *Nucl. Sci. Engr.*, **101**, 72-87 (1989).

11. G. W. Stewart, **Matrix Algorithms, Volume II: Eigensytems**. SIAM: Philadelphia (2001).

12. R. S. Modak, D. C. Sahni, and S. D. Paranjape, "Evaluation of Higher K-Eigenvalues of the Neutron Transport Equation by S_n-Method," *Annals of Nuclear Energy*, **22**, pp. 359–366 (1995).

13. R. S. Varga, **Matrix Iterative Analysis**. Prentice-Hall: Englewood Cliffs, New Jersey (1966).

14. L. A. Hageman and D. M. Young, **Applied Iterative Methods**. Academic Press: New York (1981).

15. T. A. Manteuffel, "The Tchebyshev Iteration for Nonsymmetric Linear Systems," *Numerische Mathematik*, **28**, pp. 307–327 (1977).

16. B. Fischer and R. Freund, "Chebyshev Polynomials Are Not Always Optimal," *J. Approximation Theory*, **65**, pp. 261–272 (1991).

17. O. Axelsson, **Iterative Solution Methods**. Cambridge University Press: Cambridge, England (1996).

18. G. L. Ramone, M. L. Adams, and P. F. Nowak, "A Transport Synthetic Acceleration Method for Transport Iterations," *Nucl. Sci. Engr.*, **125**, pp. 257–283 (1997).

19. R. Sanchez and S. Santandrea, "Symmetrization of the Transport Operator and Lanzcos' Iterations," in **Proceedings of the 2001 International Meeting on Mathematical Methods for Nuclear Applications**, 9-13 Sep., Salt Lake City, Utah (2001).

20. G. H. Golub and C. F. Van Loan, **Matrix Computations**. Johns Hopkins University Press, Second Edition (1989).

21. D. C. Sorensen, "Numerical Methods for Large Eigenvalue Problems," *Acta Numerica*, **11**, n. 00, pp. 519–584 (2002).

22. R. B. Lehoucq, "Implicitly Restarted Arnoldi Methods and Subspace Iteration," *SIAM J. Matrix Anal. Appl.*, **23**, n. 2, pp. 551–562 (2001).

23. M. H. Gutknecht and S. Röllin, "The Chebyshev Iteration Revisited," *Parallel Computing*, **28**, pp. 263–283 (2002).

24. J. S. Warsa, T. A. Wareing, and J. E. Morel, "Diffusion Synthetic Acceleration – Part I: Deficiencies in Multi-Dimensional Heterogeneous Problems," *Nucl. Sci. Engr.* (2002), submitted.

25. B. T. Adams and J. E. Morel, "A Two-Grid Acceleration Scheme for the Multigroup S_n Equations with Neutron Upscattering," *Nucl. Sci. Engr.*, **115**, pp. 253–264 (1993).

26. J. S. Warsa, T. A. Wareing, and J. E. Morel, "Fully Consistent Diffusion Synthetic Acceleration of Linear Discontinuous Transport Discretizations on Three-Dimensional Unstructured Meshes," *Nucl. Sci. Engr.*, **141**, pp. 236–251 (2002).

27. A. Bouras and V. Frayssé, "A Relaxation Strategy for Inexact Matrix-Vector Products for Krylov Methods," CERFACS TR/PA/00/15, European Centre for Research and Advanced Training in Scientific Computation, Toulouse, France (2000). Submitted to *SIAM J. Matrix Anal. Appl.*.

28. A. Bouras, V. Frayssé, and L. Giraud, "A Relaxation Strategy for Inner-Outer Linear Solvers in Domain Decomposition Methods," CERFACS TR/PA/00/17, European Centre for Research and Advanced Training in Scientific Computation (2000).

29.

30. V. Simoncini and D. Szyld, "Theory of Inexact Krylov Subspace Methods and Applications to Scientific Computing," Research Report 02-4-12, Temple University, Apr. (2002).

31. T. A. Wareing, E. W. Larsen, and M. L. Adams, "Diffusion Accelerated Discontinuous Finite Element Schemes for the Sn Equations in Slab and X-Y Geometries," in **Proc. International Topical Meeting on Advances in Mathematics, Computations, Reactor Physics**, 28 April - 2 May, Vol. 3, Pittsburgh, Pennsylvania, pp. 11.1 2–1 (1991).

32. G. E. Hansen and W. H. Roach, "Six and Sixteen Group Cross Sections for Fast and Intermediate Critical Assemblies," LAMS-2543, Los Alamos Scientific Laboratory, Dec. (1961).

33. G. I. Bell, J. J. Devaney, G. E. Hansen, C. B. Mills, and W. H. Roach, "Los Alamos Group-Averaged Cross Sections," LAMS-2941, Los Alamos Scientific Laboratory, Sept. (1963).

34. L. N. Trefethen and I. D. Bau, **Numerical Linear Algebra**. SIAM: Philadelphia (1997).

35. E. E. Lewis (Chairman), "Expert Group on 3-D Radiation Transport Benchmarks: Summary of Meeting," Report NEA/NSC/DOC(2001)17, OECD/NEA, Sept. (2001).

36. K. J. Maschhoff and D. C. Sorensen, "P_ARPACK: An Efficient Portable Large Scale Eigenvalue Package for Distributed Memory Parallel Architectures," in **Applied Parallel Computing in Industrial Problems and Optimization** (J. Wasniewski, J. Dongarra, K. Madsen, and D. Olesen, Eds.), Vol. 1184 of *Lecture Notes in Computer Science*, Berlin (1996), Springer-Verlag.